REMARKS

This paper is responsive to the Office Action mailed February 7, 2006. Filed herewith is a Request for a One Month Extension of Time, which extends the statutory period for response to expire on June 7, 2006. Accordingly, Applicant respectfully submits that this response is being timely filed.

Double Patenting

At ¶3-7 of the Office Action, the Examiner rejects claims 1 and 17 under obviousness-type double patenting as being unpatentable over claims 1 and 11 of copending application number 09/828,049. Accordingly, a terminal disclaimer is submitted herewith to overcome these rejections, so these rejections should be withdrawn.

Claim Rejections

At ¶8-11 of the Office Action, Applicant is encouraged to note that the Examiner has withdrawn the objection of obviousness over Aho in view of Koizumi, which was raised previously. However, the Examiner now rejects claims 1-4 and 7-18 under 35 U.S.C. §102(b) as being anticipated by US Patent No. 5,586,323 (Koizumi). The Applicant traverses these rejections for at least the following reasons.

The Examiner asserts that Koizumi anticipates claims 1-4 and 7-18. The Examiner argues that the "abstract object program" (ARM Code) described by Koizumi is equivalent to "an intermediate representation of program code" as recited in claim 1, and that the steps of "generating [in the intermediate representation] a plurality of register objects representing abstract registers" and "generating expression objects each representing a different element of said program code as that element arises in the program code, each expression object being referenced by a register object to which it relates either directly, or indirectly via references from other expression objects" of claim 1 are thus equivalent to the abstract object program containing an instruction sequence for an abstract register machine (ARM) having a plurality of registers in Koizumi. In response, the Applicant respectfully submits that this analysis of Koizumi is not correct, and that there are instead a number of substantial differences between the claimed invention and Koizumi.

1. The "abstract object program" (ARM Code) in Koizumi is not equivalent to an "intermediate representation of program code" as in the present invention

"Intermediate representation" is a term of the art and is well understood by the notional skilled person. The "abstract object program" of Koizumi is not equivalent to "intermediate representation" in the ordinary meaning of this term to the skilled person. We agree with the Examiner to the extent that the "abstract object program" is an intermediate program, but this is not equivalent to an "intermediate representation of program code" as recited in claim 1.

As discussed in the November 7, 2005 response to the second Office Action, the Applicant acknowledges that Koizumi does indeed mention "intermediate representation" referring to column 10, line 66, to column 11, line 21, and Figure 2. That is, Koizumi does indeed mention "intermediate representation" in a manner which is fully consistent with the ordinary meaning of this term to the ordinary skilled person. In Koizumi, the compiler 1001 takes as input the source program (i.e. the source program 106 in Figure 1) and produces as output the abstract object program for the Abstract Register Machine. The compiler 1001 of Koizumi uses "intermediate representation" as an intermediate step between the source program 106 and the Abstract Object Program 1007. Note that the "intermediate representation" of Koizumi is separate and distinct from the "abstract object program", and these are quite clearly different elements within the system of Koizumi. We also refer to the arguments on pages 6 and 7 of the reply dated November 7, 2005, where this point was made in detail previously.

2. Koizumi does not generate "register objects" or "expression objects" as claimed

Another major difference over Koizumi is that generating the ARM Code intermediate programming language of Koizumi does not involve the two "generating..." steps of claim 1.

The Applicant refers the Examiner to, for example, FIGs. 1 through 5, and to the associated text at paragraphs [0113] through [0120] of the application as published (US 2004/0205733A1). These figures and text describe how expression objects, abstract registers and register objects are generated in the intermediate representation in the example embodiment of the claimed invention.

The "intermediate programming language" (ARM Code) of Koizumi provides "an abstract object program 2920, which is represented by an ArmCode instruction sequence composed of instructions for the abstract register machine." (col. 12, lines 2-4). Each instruction of this ArmCode instruction sequence includes "an abstract operator 2921, an abstract register designation 2923 and an abstract memory constant designation 2925 as typical or representative items." (col. 12, lines 7-9). Thus, the "intermediate programming language" (ARM Code) taught by Koizumi is a set of instructions relating to "a model machine or computer common to many of the currently used computers" (col. 12, lines5-7), to be translated through an installer to specific machine language for execution on specific hardware. The "intermediate programming language" taught by Koizumi does not contain the relationship of expression objects, abstract registers and register objects recited in claim 1.

In particular, the second "generating..." step of claim 1 contains both procedural and structural limitations, neither of which are found in Koizumi. Note that the claim recites "generating expression objects [each representing a different element of said program code] as that element arises in the program code". Thus, the expression objects are generated at the point when the represented expression arises in the subject program code. There is a direct procedural relationship between the subject program code and the generated expression objects. This relationship does not arise in Koizumi, where instead the ARM Code is produced by the compiler (after having been through the "intermediate representation" step within the compiler). Further, in terms of structure, the expression objects are created with "each expression object being referenced by a register object to which it relates either directly, or indirectly via references from other expression objects" which results in the structure shown in the example embodiment of Figures 1-5 of the present application where the register objects R0, R1, R2 ... R5 are at the top of the diagram, and the expression objects "#3", "+" etc. are referenced directly or indirectly from the register objects. The ARM Code instructions of Koizumi do not exhibit this structure.

Since Koizumi does not teach or suggest all of the elements of claim 1, that claim should be allowable.

3. Comments concerning the dependent claims 2-15

The dependent claims 2-15 have all been discussed in detail in the Applicant's previous responses. We respectfully disagree with the Examiner's analysis of these claims for the reasons given previously. Dependent claims 2-15 all depend from allowable claim 1. Therefore those claims should also be allowable.

We invite the Examiner to look again specifically at dependent claim 3, which recites that "the register objects represent abstract registers corresponding to registers of said subject processor." That is, the method of claim 3 (dependent on claims 1 and 2) receives program code, expressed in terms of an instruction set of a subject processor, [i.e. which will be eventually translated and output as program code executable by a target computer]. Secondly, the register objects in the intermediate representation represent registers of the subject processor as referred to in the instructions of the input program code. These features are further reinforced by referring to the detailed description at paragraphs [0111] and [0112] of the present application. These features are not present in Koizumi and therefore at least claim 3 should be allowable.

At ¶12-13 of the Office Action, we are pleased to note that dependent claims 5 and 6 are acknowledged to be novel over Koizumi. However, we respectfully disagree with the Examiner's analysis concerning obviousness over Koizumi in view of Aho. The analysis of claim 5 refers to Chapter 9 of Aho (Fig 9.20, Page 559) - which, as discussed at length in Applicant's previous responses, concerns CODE GENERATION and does not concern "a method of generating an intermediate representation of program code" as claimed. Code generation comes after the intermediate representation has been generated. See Aho at page 513, first paragraph:

> "The final phase in our compiler model is the code generator. It takes as input an intermediate representation of the source program and produces as output an equivalent target program, as indicated in Figure 9.1." (emphasis added).

4. Comments concerning independent claim 16

Concerning independent method claim 16, the "abstract registers" and "abstract object program" of Koizumi are not equivalent to the "plurality of register objects" in "an intermediate representation of program code" as discussed above for claim 1. Further, the ARM Code intermediate programming language of Koizumi does not exhibit the "branched tree-like network" structure recited by claim 16, having "all register objects at the lowest basic root or tree-trunk level of the network with no register object feeding into any other register object". The ARM Code instructions illustrated in Figure 5a of Koizumi are in the format of program instructions for the hardware-independent abstract register machine. The instruction clearly includes a first operand "R1" and a second operand "R2" such that the registers R1, R2 are dependent from the instruction. It is not correct to equate the ARM Code instructions with the "intermediate representation" of claim 16, and it is not correct to equate the ARM Code instructions as being "organised into a branched tree-like network..." as claimed. Further still, the register references are clearly not at "the lowest basic root or tree-trunk level". Since Koizumi does not teach or suggest all of the elements of claim 16, that claim should be allowable.

5. Comments concerning independent claims 17 and 18

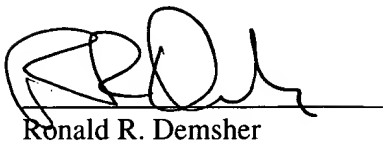
Independent claim 17 and 18 are allowable for reasons equivalent to those presented for claims 1 and 16, as discussed above. Therefore those claims should also be allowable.

In view of the above comments, applicant believes the pending application is in condition for allowance. Please charge any fees not covered and/or any credits to Deposit Account No.: 08-0219.

Should the Examiner still not feel able to issue a Notice of Allowance for any reason, the Applicant requests that the Examiner grant a personal interview. Applicant submits that a personal interview would be the most next appropriate step in order to better understand any remaining objections that the Examiner may have and to expediently resolve prosecution of this application. To arrange such an interview, please telephone the undersigned Attorney.

Respectfully submitted,

Dated: JUNE 6, 2006



Ronald R. Demsher
Registration No.: 42,478
Attorney for Applicant(s)

Wilmer Cutler Pickering Hale and Dorr LLP
60 State Street
Boston, Massachusetts 02109
(617) 526-6000 (telephone)
(617) 526-5000 (facsimile)